Guía de implementación y buenas prácticas de integración continua para el desarrollo de aplicaciones web en empresas del área metropolitana de San Salvador

María Lorena De La Fuente, William Flores, Roberto Hernández
Centro de Estudios de Postgrado,
Universidad Don Bosco,
San Salvador, El Salvador
marialo028@gmail.com, ingwilflores@gmail.com, info.redpaz@gmail.com

***Resumen*- Teniendo en consideración los beneficios que provee la Integración Continúa en la optimización de los procesos de construcción de software, como son la retroalimentación temprana, despliegue en menor tiempo y pruebas automatizadas. Se encontró la oportunidad de brindar lineamientos y mayor conocimiento de Integración Continua a las empresas del Área Metropolitana de San Salvador, que cuentan con desarrollo de aplicaciones Web y no la utilizan, para que puedan optimizar sus procesos de construcción de Software implementando Integración Continua.**

***Palabras Clave*- Arquitectura de Software, Integración Continua, DevOps, Desarrollo Ágil, Entrega Continua.**

I. INTRODUCCIÓN

Integración Continua no es un concepto nuevo, se ha vuelto cada vez un término mucho más familiar en los departamentos de desarrollo de software. De hecho, ya hay empresas en el Área Metropolitana de San Salvador que están utilizando esta práctica ágil reportando beneficios significativos.

Sin embargo, su implementación puede ser un proceso desafiante por lo que se busca generar una guía de implementación y una de buenas prácticas que sirvan de insumo para dar los primeros pasos en la construcción de un entorno de desarrollo ágil para el desarrollo de aplicaciones Web basadas en Java y C# .Net, utilizando Integración Continua. Se incluye el diseño de una arquitectura, la instalación, configuración e integración de las herramientas que conforman el entorno de IC. Así como recomendaciones para la adopción.

II. INFORMACIÓN BÁSICA DEL PROYECTO

A. Descripción del tema

El desarrollo de software es crítico para las empresas, este impulsa la innovación y aumenta la ventaja competitiva. En su estudio publicado en 2018, Freeform Dynamics encontró que la importancia del software ha crecido en todas las áreas del negocio, pero sobre todo para mejorar la línea superior donde se impulsa el crecimiento, ayuda al negocio a expandirse y competir de manera efectiva [1]. El estudio encuestó a 1,279 altos funcionarios, tanto gerentes como profesionales, en un rango de medianas a grandes empresas en 15 países, ocho sectores industriales y cinco continentes.

Acorde a las predicciones de la Consultora International Data Corporation, sobre la industria de la tecnología de la información mundial para 2018 y años posteriores se tiene que "para 2020, el 60% de las empresas se encontrarán en el proceso de implementar nuevos cimientos de Tecnología de la Información como parte de una organización totalmente articulada en torno a una estrategia de plataforma digital" [2].

La Tecnología de la Información debe responder oportunamente a las tendencias del consumidor y exigencias del negocio, lo que ha obligado a buscar optimizar los procesos de desarrollo de software. Las metodologías tradicionales han tenido que abrirle camino a la metodología ágil, la cual ha podido resolver muchos de los problemas que ha experimentado el desarrollo de software; se ha pasado de un modelo predictivo a un

modelo adaptativo, orientado a las personas y no a los procesos [3].

La metodología ágil ya no es sólo una tendencia, es una necesidad que las nuevas gerencias deben adoptar para enfrentarse al actual mercado competitivo. La brecha entre tecnología y negocio debe ser cada vez más pequeña, en conjunto definen productos mínimos viables para obtener retroalimentación temprana al estar continuamente entregando valor en periodos cortos.

El desarrollo ágil por sí solo no sería capaz de lograr entregables funcionales de manera rápida, sin el apoyo de prácticas y valores culturales que disminuyan la brecha entre desarrollo y operaciones para maximizar la velocidad de entrega de un producto o servicio, desde la idea inicial hasta producción. Este conjunto de prácticas y valores se denomina DevOps.

Una de las prácticas dentro de DevOps, que optimiza el desarrollo de software con un alto grado de calidad, pero que a su vez su adopción presenta cierto grado de complejidad es la denominada Integración Continua, identificada de aquí en adelante como IC. En su popular artículo "Continuous Integration", Martin Fowler la describe como "una práctica de desarrollo de software donde los miembros del equipo integran sus trabajos constantemente, usualmente diariamente, dando lugar a múltiples integraciones por día. Cada integración es verificada por un compilado automático, incluyendo pruebas para detectar lo más pronto posible cualquier error en la integración."

B. Planteamiento del problema

En su estudio Freeform Dynamics encontró que, aunque la mayoría de las empresas están comprometidas con la adopción tanto de DevOps como ágiles, la implementación se encuentra aún en etapas iniciales [1].

En El Salvador aunque las empresas conocen los beneficios de implementar IC, tienen la dificultad de no disponer de una fuente donde se contemplen los desafíos contextualizados a la realidad del desarrollo de software salvadoreño e incluyan buenas prácticas para obtener el máximo provecho de la IC. Inclusive en la investigación de campo se encontró que quienes han utilizado consultoría ha sido con una empresa o consultor extranjero.

En la investigación bibliográfica realizada se encontró tutoriales informales publicados en blogs enfocados a diferentes lenguajes de programación y tecnologías, cabe mencionar que la mayoría de estos se encuentran en idioma inglés.

C. Justificación

Son ampliamente conocidos los beneficios de aplicar el desarrollo ágil y prácticas de DevOps, como IC, en la optimización de los procesos de la construcción de software así brindando una ventaja competitiva a las empresas donde su estrategia se encuentra impulsada por la tecnología.

Se tiene el caso de éxito en España del banco BBVA, el cual tuvo una transformación digital adoptando metodologías ágiles para la gestión de proyectos para ganar rapidez y flexibilidad. Francisco Gonzales, presidente de BBVA, está convencido que los bancos se transformarán en empresas de software.

En 2018 la aseguradora Allstate obtuvo un reconocimiento en los premios Gartner Eye on Innovation, por la API que desarrollaron en 21 días utilizando TDD, desarrollo de software orientado a pruebas, con IC y prácticas de desarrollo ágil. Al asociarse con Uber, Allstate tuvo que revolucionar sus productos de seguros comerciales y los procesos de creación y manejo de reclamos. Sólo tenían 3 semanas para el inicio de la vigencia de la póliza y debían tener una solución para la recepción de reclamos. Formaron un equipo de negocio y tecnología para desarrollar la API que permitió integrar su sistema de reclamos con terceros; lo que ha dado camino a nuevos tipos de pólizas de automóviles comerciales y otras formas de economía compartida con modelos de negocio disruptivos.

Etsy paso de tener publicaciones del sitio completo 2 veces a la semana que tardaban 4 horas a un enfoque más ágil. InfoQ, en el artículo publicado en 2014, amplía como Etsy ha logrado por medio de la IC, junto a otras prácticas de entrega continua, tener 50 implementaciones al día y más de 14,000 series de pruebas con 150 desarrolladores. [4]

La Consultora International Data Corporation en sus predicciones indica que se acelerará el ritmo de innovación debido a la adopción de prácticas de DevOps, como IC, que permiten aumentar el número de lanzamientos anuales de código de aplicación en un 50%. [5]

En el artículo en línea de Nearshore Americas, el principal recurso de investigación, datos y noticias para tomadores de decisiones de negocios que participan en el suministro de servicios de conocimiento e inversión en América Latina, el Caribe y Canadá, indican que las empresas están aprendiendo a tomar ventaja de las nuevas tecnologías apoyándose del desarrollo ágil para generar valor al negocio y esto está generando una disrupción dentro de la industria de Tecnología de Información.

Antes las empresas norteamericanas buscaban adquirir sus desarrollos de software en la India, sin embargo, el desarrollo ágil necesita de constante comunicación entre tecnología y el negocio, sobre todo para la resolución de bugs; colocando en desventaja a la India y brindándole una ventaja por la cercanía de horario a las empresas de Latinoamérica [6].

Esta ventaja competitiva puede ser aprovechada por las empresas salvadoreñas que se dedican al desarrollo de software, al encontrarse listas aplicando desarrollo ágil y prácticas de DevOps, que les permitan optimizar sus procesos de construcción de software.

Al entregar una guía de implementación, se está apoyando a las empresas salvadoreñas a dar los primeros pasos a montarse en un entorno ágil con IC y así estar preparados para ser competitivos en el mercado.

D. Objetivos

1) Objetivo general

La investigación tiene como objetivo diseñar una guía de implementación y buenas prácticas de IC para el desarrollo de aplicaciones Web en empresas del Área Metropolitana de San Salvador, AMSS, basada en la experiencia de las empresas de la misma área y expertos en el campo de IC. Se pretende sea una referencia para empresas que desarrollan aplicaciones Web y tengan el interés de implementar IC para optimizar sus procesos de construcción de software.

2) Objetivos específicos

- Identificar los desafíos más comunes durante la implementación de IC en las empresas en estudio del AMSS.
- Identificar las herramientas más utilizadas en un entorno de IC para las actividades de desarrollo de aplicaciones Web.
- Diseñar una guía de buenas prácticas en las actividades de IC para el desarrollo de aplicaciones Web.
- Diseñar una guía de implementación de IC para el desarrollo de aplicaciones Web, enfocada al entorno de desarrollo Java y .NET que la mayoría de las empresas en estudio del AMSS utilizan.

III. CONCEPTOS FUNDAMENTALES

El primero es DevOps, el cual es una cultura, es la unión de personas, procesos y tecnologías para permitir la entrega continua de valor a los clientes. Busca establecer una cultura de colaboración entre desarrollo y operaciones que tradicionalmente trabajaban asilados.

Siendo parte de las prácticas ágiles de DevOps, tenemos Integración Continua, Fowler la define como una práctica de desarrollo de software donde los miembros del equipo integran sus trabajos constantemente [7].

La Integración Continua consiste en que los desarrolladores versionan sus cambios de forma periódica en un repositorio compartido con un sistema de control de versiones. Antes se ejecutan pruebas unitarias para verificar y validar el código previo a la integración. Se integra y genera el compilado para automáticamente aplicarle pruebas, por ejemplo de análisis de código estático. El objetivo clave es detectar y arreglar errores con mayor rapidez, mejorar la calidad del código y reducir el tiempo de salida a producción.

Para entender el proceso de IC, es necesario conocer donde inicia realmente dicho proceso. Todo inicia cuando el desarrollador hace confirmaciones de su código fuente a un repositorio. En proyectos donde hay muchas personas con distintos roles pueden hacer cambios que inician el ciclo de IC. Los desarrolladores cambian el código fuente, el o los administradores de base de datos cambian la definición de tablas, los

equipos construyen y despliegan cambios en los archivos de configuración, interfaces, etc.

1) Repositorio de Control de Versiones

El propósito de un repositorio de control de versiones es administrar los cambios del código fuente y otros activos de software, como por ejemplo la documentación, utilizando un repositorio de acceso controlado.

2) Servidor de Integración

Un servidor de integración ejecuta una compilación de integración cada vez que se realiza cada vez que se realiza cambios en el repositorio de control de versiones. Normalmente, se configura el servidor de IC para verificar cambios en el repositorio cada pocos minutos o más. El servidor de integración recibe los archivos fuentes y ejecuta un script o varios scripts de compilación.

3) Script de Construcción

Un script de construcción básicamente es un simple script o conjunto de scripts que se pueden usar para compilar, realizar pruebas, inspeccionar y desplegar software. Podríamos implementar un script de construcción sin implementar un sistema de IC. Ant, NAnt,make, MSBuild son algunos ejemplos de herramientas de construcción que pueden automatizar la construcción del ciclo de IC.

4) Servidor de Pruebas

Servidor cuyo objetivo es permitir la realización de la automatización de pruebas.

IV. ENTORNO DE INTEGRACIÓN CONTINUA

La implementación de integración continúa tanto en un entorno de desarrollo web basado en .NET y el otro en Java permite abarcar los entornos más utilizados y demandados según el estudio realizado.

Para comenzar se exponen las arquitecturas utilizadas para cada uno de ellos. De un vistazo se puede dar cuenta de las herramientas necesarias en cada uno de ellos: servidor de integración, de versionamiento, de pruebas, compiladores, etc.

1) Arquitectura .Net

La arquitectura para aplicaciones basadas en .NET ha sido diseñada especificando todas aquellas herramientas indispensables para dicho entorno. Jenkins como responsable del proceso de integración, la gestión de artefactos está bajo el control de JFrog Artefactory, GitLab como administrador de repositorios, Internet Information Service (IIS) es el servidor web, SonarQube y Selenium como servidores de pruebas

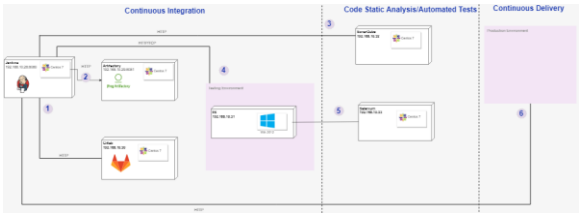

Imagen 1: Arquitectura de entorno IC para .NET

2) Arquitectura Java

Del lado de aplicaciones basadas en el lenguaje Java se ha construido una arquitectura de IC , la cual en realidad, usa las mismas herramientas descritas en la sección de .NET con la diferencia que las aplicaciones Java necesitan del servidor de artefactos Nexus en vez de JFrog Artifactory y un servidor web con Tomcat.

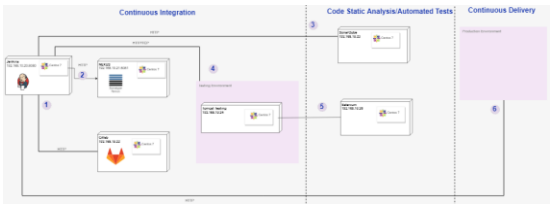

Imagen 2: Arquitectura de entorno IC para Java

3) *Implementación del entorno de IC*

Para poder llevar a cabo una implementación de un entorno de IC es importante describir todas aquellas herramientas necesarias para ello. La primera de ellas es un sistema de control de versiones el cual permite controlar y dar seguimiento a todos los cambios realizados en el código fuente, así se tiene la capacidad de volver a un estado anterior del código. En este estudio se decidió usar Git por ser distribuido, adecuado para pequeños y grandes proyectos y de código libre. Pero además, es necesario un administrador de repositorio. GitLab, ofrece la capacidad de gestionar repositorio Git, creación de wiki y seguimiento de problemas.

Además, un entorno de IC requiere disponer de un servidor de integración; el más usado por su compatibilidad, como principal característica es Jenkins. Es el responsable de integrar el código y ejecutar el "pipeline" definido, que va desde obtener las actualizaciones del código desde el administrador de repositorio hasta compilar y ejecutar las diferentes pruebas establecidas.

Para inspección continua de la calidad del código y realizar revisiones automáticas con análisis de código estático y así detectar errores y vulnerabilidades de seguridad es necesario un servidor de pruebas como SonarQube. Y para la realización de pruebas funcionales se requiere de un marco de referencia como Selenium.

También el entorno de IC se apoya de una herramienta que permite la construcción, gestión de componentes y artefactos, para ello se puede cumplir con esa tarea se puede usar Nexus para entornos Java y JFrog Artifactory para .NET. Ambos tienen la capacidad de administrar piezas de software, binarios y sus dependencias.

Por último es necesario un lugar donde las aplicaciones construidas bajo un entorno IC puedan residir y exponerse. El servidor de despliegue es el lugar donde dichas aplicaciones estarán disponibles después de haber pasado por todo el flujo de IC. Para el caso de aplicaciones Java se requiere de un servidor web como Tomcat e Internet Information Service (IIS) para las aplicaciones .NET.

1) Implementación

El proceso de implementación de IC comienza con la instalación y configuración inicial de todas las herramientas. La primera de ellas es el servidor de integración, que para este estudio ha sido Jenkins. Este servidor juega el papel de orquestador y responsable de ejecutar el "pipeline" de IC. Para su funcionamiento basta con instalarlo y realizar algunas configuraciones iniciales como: creación de usuario y nombre de dominio.

La segunda herramienta a instalar y configurar es Gitlab, el servidor de administración de repositorios Git. Su configuración inicial requiere de la creación de perfiles de usuarios, grupos y permisos.

Dentro del flujo de IC entra en escena el análisis de código estático para detectar vulnerabilidades, función bajo responsabilidad de SonarQube, el servidor de pruebas.

Además, dependiendo del entorno que se esté utilizando, el siguiente componente a instalar y configurar es el servidor de artefactos, Nexus para aplicaciones Java y Jfrog Artifactory para .NET. La configuración inicial implica la configuración de usuario, establecimiento del directorio de datos y crear las entradas para administrar Nexus como servicio.

Dado que ya se tienen las herramientas instaladas y configuradas es el momento de integrarlas. En otras palabras, hay que enlazar cada una de ellas con el orquestador de IC; Jenkins. Pero primero es necesario tener completa certeza de la comunicación existente a nivel de red y de puertos de los servidores.

La integración comienza con el repositorio de artefactos Nexus si son aplicaciones Java o Jfrog Artifactory si son .NET y Jenkins el servidor para la construcción automatizad. Para ello es necesario crear el repositorio respectivo, configurar usuarios, roles y permisos. De lado de Jenkins se descargan e instalan los complementos necesarios.

Integrar el administrador de repositorios (GitLab) y el servidor de integración (Jenkins) es el siguiente paso. De lado de GitLab, se realiza la configuración inicial y se instalan todos los complementos necesarios para que GitLab funcione correctamente y de forma integrada con Jenkins. En este paso también se crea el "pipeline" que ejecutara Jenkins, que no es más que los diferentes pasos a realizar como lo son: servir de escuchante "listener" cuando hay "push" en GitLab, construir el proyecto, ejecución de pruebas y despliegue.

Ahora es momento de integrar la automatización de pruebas con el servidor de integración. En este punto primero se busca integrar Jenkins con el servidor web Tomcat, el cual sirve como servidor de despliegue de la aplicación. Para funcionar correctamente se instalan y configuran los complementos requeridos para cada uno de ellos.

El último paso en el proceso de implementación de un entorno de IC es la integración entre Jenkins y el servidor de pruebas. La idea es que Jenkins ejecute las pruebas unitarias que podrían estar construidas con Junit para el caso de Java y NUnit para .NET, además se configuran las herramientas de automatización de pruebas funcionales y de regresión con Selenium Framework.

VI. CONCLUSIONES Y RECOMENDACIONES

La implementación del entorno de IC puede ser un proceso desafiante por varias razones, se encontró que principalmente por la cultura organizacional y adopción de las prácticas por los equipos de desarrollo.

Como apoyo para afrontar los desafíos, se diseñó la arquitectura de cada entorno de IC, C# .Net y Java, seleccionando las herramientas en base a la recomendación de las empresas y expertos que consideraron la realidad de las empresas salvadoreñas para comenzar a utilizar IC.

Así también se crearon las guías de implementación que exponen la instalación, configuración e integración de las herramientas. Se detallaron los pasos a seguir para montar el entorno de IC, ya sea local o en la nube.

Adicional se recolectó buenas prácticas para la implementación de IC en base a la experiencia de las empresas del Área Metropolitana de San Salvador, que han implementado y han estado utilizándola por lo menos durante 6 meses en sus desarrollos de aplicaciones Web. Además de incorporar recomendaciones de expertos en el área de IC bajo entornos de desarrollo Web.

Se pretende que esto sea un insumo para las empresas interesadas en implementar un entorno de desarrollo ágil para aplicaciones Web utilizando IC.

Como un resumen, se exponen las buenas prácticas recopiladas:

1. Desarrollo ágil: La Integración y Entrega Continua están alineados a los principios del desarrollo ágil, cómo es el de la entrega continua, “Entregamos software funcional frecuentemente, entre dos semanas y un mes, con preferencia por periodos de tiempo lo más corto posibles”.

   Las empresas en estudio identificaron la relación de integración y entrega continua con el desarrollo ágil, por lo que quienes se encontraban utilizando modelos tradicionales, implementaron modelos ágiles para aprovechar los beneficios de la integración y entrega continua; siendo Scrum el modelo más utilizado. Este es un cambio organizacional, por lo que requiere el apoyo de la alta gerencia, así como la capacitación y definición de nuevos roles dentro de equipos multidisciplinarios.

2. Adopción: Un cambio organizacional, ya sea que se trate de agregar un nuevo proceso o modificar uno existente, puede generar resistencia por lo que al iniciar un proceso de implementación de IC debe existir una correcta gestión del cambio.

   Se ha coincidido con las empresas y expertos que la gestión del cambio y la cultura ágil son claves para la adopción de IC. Se debe invertir más tiempo en conocer a los equipos, su experiencia y conocimiento, para poder hacerlos parte del nuevo proceso que en la selección y configuración de herramientas. Se busca crear una cultura organizacional de IC que abrace el cambio y la innovación.

   Lianping Chen [8], presenta estrategias para superar los retos de adopción de integración y entrega continua, como son venderlo como un analgésico, tener un equipo dedicado multidisciplinario, entrega continua de entrega continua, iniciar con aplicaciones sencillas pero importantes, así como tener un esqueleto visual del pipeline.

3. Definir las aplicaciones que serán migradas al entorno de IC: Así se define el alcance del proyecto de implementación en cuanto al catálogo de aplicaciones que serán parte de dicho cambio. Esto dará una ruta crítica en cómo avanzar para lograr que dichas aplicaciones puedan utilizarse bajo un entorno de IC.

4. Apoyarse de expertos en IC que facilite y apoye en el proceso: Es un buen ejercicio referenciarse con otras empresas que ya estén utilizando IC en sus desarrollos, sí se tiene filiales o empresas hermanas se pueden apoyar con la experiencia de las mismas. Esto servirá para tener un mejor contexto antes de iniciar.

   Es recomendable buscar apoyo con expertos en IC, esto puede ahorrar tiempo por el conocimiento y orientación que puedan aportar en la implementación. Sin embargo, no todas las implementaciones han sido apoyadas por expertos o consultores desde el inicio, muchas han sido iniciativas de Tecnología y por el perfil de los miembros del equipo han podido implementar un primer entorno de IC.

5. Gestión de cuentas centralizada: Debido a la cantidad de herramientas a utilizar, es recomendable manejar un solo repositorio de cuentas ya sea por base de datos, LDAP u otro contenedor, ya que de esta manera hay un mayor control y orden sobre la información.

6. Selección de herramientas: Algunas de las herramientas utilizadas en un entorno de IC ya cuentan con características necesarias para implementarlas por completo sin la necesidad de depender de otras herramientas que permitan sostener dicho entorno. Se sugiere que se implemente el ambiente de IC con herramientas que ya cuentan con esas características y así evitar la complejidad de mantener diferentes herramientas. Por ejemplo Jenkins es un servidor de integración que provee la mayoría de características necesarias para todo lo que se necesita de IC.

   Además se hace especial énfasis en los criterios para seleccionar las herramientas. Uno de ellos es considerar el soporte, así como también estar consciente sobre el lenguaje de programación que se tiene en ese momento y la facilidad que puede tener de integración con las demás herramientas.

   No es conveniente para la implementación que dichas herramientas sean de alta complejidad y de baja flexibilidad, ya que el estar a la vanguardia o seguir el ritmo del crecimiento y evolución del negocio podría resultar en un alto coste económico y de tiempo. Sería más oportuno contar con herramientas de uso fácil y sencillo, y que además soporten múltiples plataformas, así como que no provoquen ataduras tecnológicas considerables.

   En base a la realidad salvadoreña de las empresas, se ha encontrado una preferencia notable a las herramientas de código abierto y al ser integradas de manera interna por sobre herramientas complejas y costosas, que podrían ser en un principio subutilizadas. Así también esto permite ir completando el flujo de IC conforme los equipos de desarrollo se van adaptando a las prácticas de IC y estándares de código.

7. Estrategia de ramas y versionamiento: Es recomendable definir la estrategia, norma o política de cómo nombrar las ramas, los “commit” y los lanzamientos, por ejemplo los lanzamientos podrían llevar como prefijo “v” seguido por el número de versión (v1.1.0); los “commit” deberían incluir información sobre las funcionales o cambios que se están incorporando; también la política de ramas es recomendable basarse en estrategias estrictas y validadas como por ejemplo las que define GitFlow.

8. Integraciones y gestión de conflictos: IC basa su filosofía en poder hacer que los equipos integren su código diariamente. Así que, es una buena práctica cuando se inicia un proceso de implementación definir la frecuencia con la que se harán tales integraciones. Si se quiere que un entorno de IC trabaje efectivamente, los desarrolladores deberán cambiar sus hábitos de desarrollos de software, esta estrategia los obliga a realizar integraciones, “pull” y “push”, a la rama de desarrollo diariamente o cuando se tenga un componente terminado.

   Además es importante definir una estrategia para la resolución de conflictos que seguramente se van a presentar cuando se realizan las integraciones. Se puede configurar y ajustar Git para que siga una estrategia predefinida cuando se

hagan las integraciones. Es recomendable configurar notificaciones cuando suceden errores.

9. Estrategia de pipeline: El pipeline define cómo será el flujo que seguirá todo el proceso de IC, se debe dedicar tiempo para definir una estrategia que permita asegurar que el proceso es el más óptimo para el proyecto, basado en las necesidades y objetivos de la empresa.

10. Pruebas: Es recomendable el uso de pruebas unitarias, funcionales y de regresión automatizadas, porque mitigan el riesgo de errores como consecuencia de un cambio.

11. Métricas. Desde el inicio de la implementación se deben de establecer y priorizar las reglas, métricas y aspectos que se necesitan medir según el sentido de las metas del negocio, de lo contrario el resultado de la evaluación podría quedar sub o sobre estimado, claro ejemplo es el uso de Selenium o SonarQube, para los cuales, sí se seleccionan las reglas que van mayormente alineadas con los objetivos de la empresa, la evaluación tendrá más efectividad al momento de medir la calidad del código y funcionamiento de la aplicación, y con ello se podrán atacar las vulnerabilidades o fallos de manera priorizada y eficaz.

REFERENCIAS

[1] CA Tecnologies. (10 de 2018). CA Tecnologies. Obtenido de CA Tecnologies: https://www.ca.com/us/modern-software-factory/content/how-agile-and-devops-enable-digital-readiness-and-transformation.html#formanchor

[2] Torrejon, L. (07 de noviembre de 2017). Las predicciones de IDC proporcionan un modelo y las bases para convertirse en una empresa nativa digital. Obtenido de http://www.blog-idcspain.com/predicciones-idc/.

[3] Navarro Cadavid, A., Fernández Martínez, J. D., & Morales Vélez, J. (2013). Revisión de metodologías ágiles para el desarrollo de software. Prospect. Vol. 11, No. 2, 30-39.

[4] Miranda, João. (17 de marzo de 2014). www.infoq.com. Recuperado el 27 de noviembre de 2016, de www.infoq.com: https://www.infoq.com/news/2014/03/etsy-deploy-50-times-a-day.

[5] IDG. (2016, 3 18). IDG Connect. Retrieved from IDG Connect: https://www.idgconnect.com/idgconnect/opinion/1012700/devops-changing-role

[6] Patrick, H. (2015, 12 10). Nearshore Americas. Retrieved from Nearshore Americas: https://www.nearshoreamericas.com/devops-transformation-boom-latin-america-bust-india/

[7] Duvall, P. M., Matyas, S., & Glover, A. (2008). Continuous Integration, Improving Software Quality and Reducing Risk. Dorling Kindersley.

[8] Elsevier Inc. (2017). Continuous Delivery: Overcoming adoption challenges. L. Chen / The Journal of Systems and Software, 72-86.

[9] Amazon.com, Inc. (28 de Diciembre de 2018). Explicación de la integración continua. Obtenido de Amazon Web Services, Inc.: https://aws.amazon.com/es/devops/continuous-integration/

[10] Arias, F. G. (1999). El Proyecto de Investigación. 3ra edición. Caracas: Episteme. Oral Ediciones.

[11] Aumaille, Benjamín. (Noviembre 2002), J2EE Desarrollo de aplicaciones Web, Barcelona, España: Ediciones ENI.

[12] Atlassian Confluence 6.12.2. (24 de Diciembre de 2018). JFrog Become a Professional! Obtenido de Installing Artifactory on Windows: https://www.jfrog.com/confluence/display/RTF/Installing+on+Windows.

[13] Berna Zipa, M. M. (2015). Gestión por Procesos y Mejora Continua, Puntos Clave para la Satisfacción del Cliente. Bogotá: Universidad Militar Nueva Granada.

[14] Bioul, G., Escobar, F., Alvarez, M., Nardin, A., & Ricci Aparicio, E. (2010). Metodologías Ágiles, análisis de su implementación y nuevas propuestas. CACIC 2010 - XVI Congreso Argentino de Ciencias de la Computación, 597-606.

[15] Business Wire. (14 de 10 de 2018). Business Wire. Obtenido de Business Wire: https://www.businesswire.com/news/home/20181014005052/en/Gartner-Announces-Winners-2018-Gartner-Eye-Innovation.

[16] CA Tecnologies. (10 de 2018). CA Tecnologies. Obtenido de CA Tecnologies: https://www.ca.com/us/modern-software-factory/content/how-agile-and-devops-enable-digital-readiness-and-transformation.html#formanchor.

[17] Casal, J., & Mateu, E. (2003). TIPOS DE MUESTREO. Revista Epidem. Med. Prev, 1, 3-7.

[18] Casteleyn, S. et al. (2009). Engineering Web Applications. Springer-Verlag Berlin Heidelberg.

[19] Clareburt, J. (12 de 2017). Github. Obtenido de Github: https://github.com/aspnet/Tooling/blob/AspNetVMs/docs/create-asp-net-vm-with-webdeploy.md.

[20] Código de Comercio (2008). Corte Suprema de Justicia de El Salvador Centro de Documentación Judicial, de www.redhat.com.

[21] Corona, B., Muñoz, M., Miramontes, J., Calvo-Manzan, J. A., & San Feliu, T. (Abril de 2016). Recibe, Revista Electrónica. Obtenido de Estado de arte sobre métodos de evaluación de metodologías ágiles en las pymes: http://recibe.cucei.udg.mx/revista/es/vol5-no1/computacion03.html.

[22] Directorio de Clientes que utilizan IC con CircleCI. circleci.com. Recuperado el 27 de Noviembre de 2016, circleci.com: https://circleci.com/customers/

[23] Forsgren, N., & Humble, J. (27 de Octubre de 2015). The Role of Continuous Delivery in it and Organizational Performance. Obtenido de In the Proceedings of the Western Decision Sciences Institute (WDSI) 2016, Las Vegas, NV: https://ssrn.com/abstract=2681909.

[24] Fowler, M. (01 mayo de 2006). MartinFowler.com. Recuperado 20 de noviembre del 2016, de martinfowler.com: http://www.martinfowler.com/articles/continuousIntegration.html.

[25] Forrester Consulting. (Marzo de 2013). info.thoughtworks.com. Obtenido de Continuous Delivery: A Maturity Assessment Model: http://info.thoughtworks.com/rs/thoughtworks2/images/continuous_delivery_a_maturity_assessment_modelfinal.pdf

[26] García Orozco, A. M. (2015). La Integración Continua y su Aporte al Aseguramiento de la Calidad en el Ciclo de Vida del Desarrollo de Software. Bogotá: Universidad Distrital Francisco José de Caldas.

[27] Hernandez, R. (5 de 7 de 2015). Testeando Software. Obtenido de Testeando Software: https://testeandosoftware.com/sonarqube-instalacion-basica/.

[28] Honig, R. (2015). Best Practices Integration Testing, Recuperado el 11 de 2016, de http://techbeacon.com/6-best-practices-integration-testing-ci-environment.

[29] Howtoforge. (1 de Octubre de 2018). Howtoforge Linux Tutorials. Obtenido de How to Install and Configure GitLab CE on CentOS 7:

https://www.howtoforge.com/tutorial/how-to-install-and-configure-gitlab-ce-on-centos-7/.

[30] HugeServer. (5 de Noviembre de 2018). HugeServer Knowledgebase. Obtenido de How to Install GitLab on CentOS 7: https://www.hugeserver.com/kb/how-install-gitlab-centos7/

[31] IDG Network. (02 de Noviembre de 2016). Computerworld from IDG. Obtenido de Las 10 principales predicciones de la industria TIC: http://www.computerworld.es/tendencias/las-10-principales-predicciones-de-la-industria-tic.

[32] Joshi, A. (02 de 08 de 2012). Microsoft. Obtenido de Microsoft: https://docs.microsoft.com/en-us/iis/publish/troubleshooting-web-deploy/troubleshooting-web-deploy-problems-with-visual-studio

[33] Jummp. (20 de 5 de 2010). Jummp. Obtenido de Jummp: https://jummp.wordpress.com/2010/05/20/reflexiones-sobre-la-adaptacion-de-sonar-a-nuestra-metodologia-de-calidad-para-su-uso-en-mi-organizacion/.

[34] Kili, A. (25 de Octubre de 201). Tecmint: Linux Howtos, Tutorials & Guides. Obtenido de How to Fix “firewall-cmd: command not found” Error in RHEL/CentOS 7: https://www.tecmint.com/fix-firewall-cmd-command-not-found-error/.

[35] KPMG Ltda. (1 de Diciembre de 2018). home.kpmg. Obtenido de Gestión del Cambio: https://home.kpmg/co/es/home/services/advisory/management-consulting/capitalhumano-y-gestion-del-cambio/gestion-del-cambio.html

[36] Knowledge Powerhouse (2016). Microservices Interview Questions: Good Collection of Questions Faced in Architect Level Technical. Edición Digital.

[37] Microsoft Corporation. (5 de Enero de 2017). Microsoft Hardware Dev Center. Obtenido de Enable .NET Framework 3.5 by using the Add Roles and Features Wizard: https://docs.microsoft.com/en-us/windows-hardware/manufacture/desktop/enable-net-framework-35-by-using-the-add-roles-and-features-wizard

[38] Microsoft Corporation. (26 de Octubre de 2017). Windows Support. Obtenido de Error 0x80070057 when you format a hard disk drive to install Windows 7: https://support.microsoft.com/en-hk/help/2476568/error-0x80070057-when-you-format-a-hard-disk-drive-to-install-windows.

[39] Microsoft Corporation. (1 de Septiembre de 2018). An introduction to NuGet. Obtenido de Microsoft Docs: https://docs.microsoft.com/en-us/nuget/what-is-nuget.

[40] Martínez Miguélez, M. (2006). La Investigación Cualitativa (Síntesis Conceptual). Revista Invest.en Psicol., 9(1), 123-146.

[41] Martínez-Salgado, C. (2012). El muestreo en investigación cualitativa. Principios básicos y algunas controversias. Ciência & Saúde Coletiva, 17(3), 613-619.

[42] Moquillaza Henríquez, S. D., Vega Huerta, H., & Guerra Grados, L. (Diciembre de 2010). Programación en N capas. Revista de Investigación de Sistemas e Informática, 57-67.

[43] Null, C. (12 de 2017). Tech Beacon. Obtenido de Tech Beacon: https://techbeacon.com/10-companies-killing-it-devops.

[44] Oliveros, A., del Valle Rojo, S., Wehbe, R., & Rousselot, J. (2011). Requerimientos para Aplicaciones Web. XIII Workshop de Investigadores en Ciencias de la Computación, 577-582.

[45] Palacio, A. L. (2015). Evaluación del grado de agilismo basado en los objetivos y necesidades de los equipos de trabajo. Valencia: Universidad Politécnica de Valencia.

[46] Pinzón, S., Carlos, J., & Bolaños, G. (junio de 2008). La gestión, los procesos y las metodologías de desarrollo de software. Actualidad Tecnológica, 83-98.

[47] Pressman, R. S. (2010). Ingeniería del Software: Un enfoque práctico (Séptima ed.). New York: McGraw-Hill.

[48] Randell, B. (1996). The 1968/69 NATO, Software Engineering Reports. Dagstuhl: Universidad de Newcastle.

[49] Real Academia Española. (2014). Continuo. En Diccionario de la lengua española (23.a ed.). Recuperado de http://dle.rae.es/?id=AVqx1Ac.

[50] Red Hat, Inc. Red Hat, Red Hat Enterprise Linux (2012) IC con JBoss Trading, una aplicación empresarial de referencia. JBOSS Enterprise Middleware, de www.redhat.com.

[51] Sampieri, R., Collado, C., & Lucio, P. (2010). Metodología de la Investigación. México DF: McGRAW-HILL / INTERAMERICANA EDITORES, S.A. DE C.V.

[52] Sayed, A. (13 de 7 de 2018). Tech Beacon. Obtenido de Tech Beacon: https://techbeacon.com/why-agile-devops-are-key-any-digital-transformation.

[53] Setende, Hamilton. (2012). www.academia.edu. Recuperado el 23 de noviembre del 2016, de www.academia.edu: http://www.academia.edu/4865003/Differences_advantages_and_disadvantages_between_in-house_development_IT_systems_and_industry_standard_ERP_system.

[54] Software in the Public Interest. (Diciembre de 2018). Jenkins User Documentation. Obtenido de https://jenkins.io/doc/.

[55] Sommerville, I. F. (2005). Ingeniería del Software. / Ian Sommerville. Madrid, España: PEARSON.

[56] Techopedia. (2016). Build. En Enciclopedia Digital Techopedia. Recuperado de https://www.techopedia.com/definition/3759/build.

[57] TIOBE software BV. (febrero de 2018). TIOBE The Software Quality Company. Obtenido de https://www.tiobe.com/tiobe-index/.

[58] Uchino, M. (12 de 2016). Matthew Yuki Uchino. Obtenido de Matthew Yuki Uchino: http://matthewyukiuchino.com/how-to-use-jenkins-to-deploy-c-web-applications-to-iis/.

[59] Unidad Editorial Información General, S. (22 de 12 de 2015). Unidad Editorial. Obtenido de Unidad Editorial: http://www.expansion.com/economia-digital/companias/2015/12/22/56795576268e3ea2388b471f.html.

[60] Universidad Tecnológica del Suroeste de Guanajuato. (2013). Ciencias de la Ingeniería y Tecnología, Handbook T-I. Guanajuato: ECORFAN.

[61] Vaca Barahonaa, B., Hidalgo Poncea, B., & Arcos Medinaa, G. (2015). La gestión y la producción de software con plataformas tecnológicas colaborativas. Revista Tecnológica ESPOL, 105-120.

[62] Yepes González, J. D., Pardo Calvache, C. J., & Gómez Gómez, O. S. (2015). Revisión sistemática acerca de la implementación de metodologías ágiles y otras modelos en micro, pequeñas y medianas empresas de

software. Revista Tecnológica ESPOL – RTE, Vol. 28, N. 5, 464-479.